(12) UK Patent Application (19) GB (11) 2 283 152 (13) A

(43) Date of A Publication 26.04.1995

(21) Application No 9321527.5

(22) Date of Filing 19.10.1993

(71) Applicant(s)
International Business Machines Corporation

(Incorporated in USA - New York)

Armonk, New York 10504, United States of America

(72) Inventor(s)
Keith Robert Barraclough
Adrian Charles Gay

(74) Agent and/or Address for Service
Roger James Burt
IBM UK Ltd, Intellectual Property Dept.,
Mailpoint 110, Hursley Park, WINCHESTER,
Hampshire, SO21 2JN, United Kingdom

(51) INT CL[6]
H04M 9/02 , H04L 12/433

(52) UK CL (Edition N)
H4K KTP
H4P PPBB PPEC

(56) Documents Cited
GB 2207581 A EP 0130431 A1 US 5127001 A

(58) Field of Search
UK CL (Edition L) H4K KFT KTK KTP , H4P PPBB PPEC
INT CL[5] H04L , H04M

(54) Audio transmission over a computer network

(57) A computer workstation includes an audio adapter card for generating a sequence of digital audio data samples, aid accumulating them into audio data blocks. These are then transferred one at a time into a queue in the main memory of the computer workstation. A first program loop on the workstation receives an interrupt from the audio adapter card indicating the transfer of another audio data block, and maintains a record of the head of the queue. Another program loop requests access to the computer network, transmits messages from the workstation, and maintains a record of the tail of the queue. Each audio packet transmitted from the workstation incorporates essentially all the audio data currently enqueued.

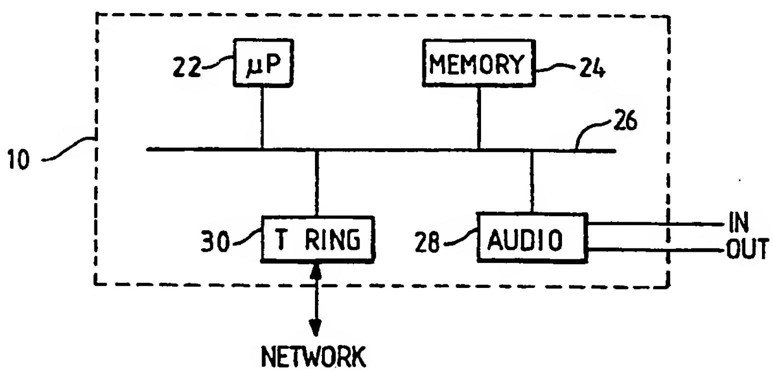


FIG. 1

GB 2 283 152 A

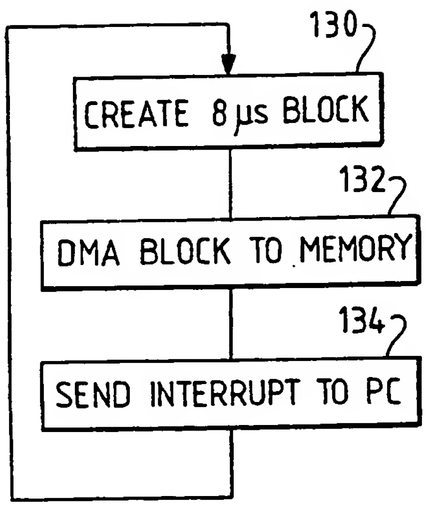


FIG. 4

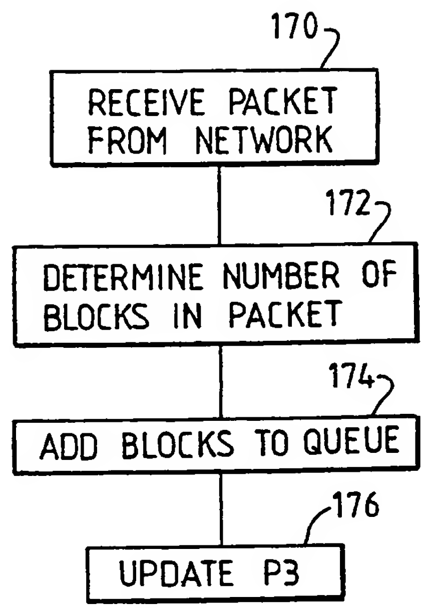
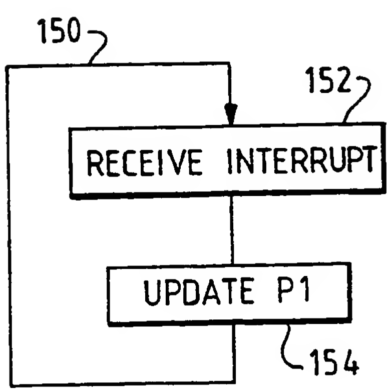
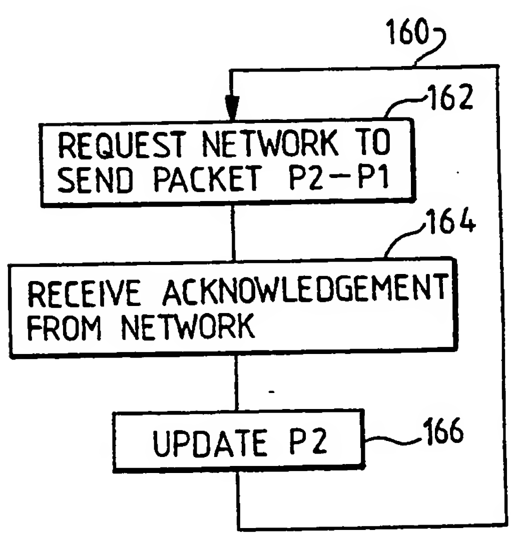


FIG. 6

FIG. 5





echos, and more importantly, can render natural interactive two-way conversation difficult (in the same way that an excessive delay on a transatlantic conventional phone call can be highly intrusive).

The above-mentioned article by Ravasio et al describes the use of a buffer of up to three packets at both the transmitting and receiving ends. If the buffers are about to overflow, because there is a delay in sending or several packets arrive close together, older packets are discarded. This leads to some gaps in the playout signal, but these can be filled by interpolation or silence (the human ear is relatively tolerant of such interference). A slightly more sophisticated approach at the receiving station is described in "Adaptive Audio Playout Algorithm for Shared Packet Networks", by B Aldred, R Bowater, and S Woodman, IBM Technical Disclosure Bulletin, p 255-257, Vol 36 No 4, April 1993. Again, packets that arrive with more than a maximum allowed delay are discarded. The amount of buffering however is adaptively controlled depending on the number of discarded packets (any other appropriate measure of lateness could be used). If the number of discarded packets is high, the degree of buffering is increased, whilst if the number of disarded packets is low, the degree of buffering is decreased. The size of the buffer is altered by temporarily changing the play-out rate (this affects the pitch; a less noticeable technique would be to detect periods of silence and artificially increase or decrease them as appropriate).

An adaptive buffering system is also disclosed in US 5127001 for the reception of multiple channels. In this system the buffer queue length is monitored, and the playout frequency varied in accordance with buffer occupancy. It is suggested in US 5127001 that the transmission queue can be linked to the same frequency, which helps to provide synchronisation over the network. The use of elastic buffers at the transmitting and receiving stations is also described in US 4866704, in this case with particular reference to compensating for slight differences in the clock rates at different nodes.

An additional source of delay in voice communications over a LAN necessarily results from the accumulation of voice samples into data packets. Thus, if the data packet represents say 32 ms of voice samples, the earliest sample is nearly 32 ms old by the time that the packet is

obtaining access to the network at the first workstation; and

transmitting a packet containing essentially all the digital audio samples that are currently in the queue each time access is obtained to the network.

Such an approach recognises that packet size, which has been been the focus of much attention in the prior art, is not necessarily the most important parameter in audio transmission over the network. In some circumstances, particularly at periods of high network utilisation, communications can be disrupted by long delays in obtaining access to the network. Such delays are extremely damaging to voice signals, which require low latency in order to maintain acceptable quality. Therefore, whenever network access is achieved, all the available data is transmitted, rather than just a single fixed packet, as in the prior art. Furthermore, the ability to transmit essentially all the enqueued data in a single packet obviates the need for deliberate buffering at the transmitting terminal, reducing the overall transmission delay.

Preferably the method further comprises the step of maintaining information indicating the number of digital audio samples in the queue, and updating said information each time a new digital audio sample is added to the queue or digital audio samples are transmitted from the queue. This allows the appropriate packet size for the next transmission to be readily calculated.

Typically the digital audio samples are generated at a constant frequency, and a predetermined number of digital audio samples are accumulated into blocks of audio data, and each transmitted packet comprises an integral number of blocks of audio data. The digital audio samples are stored in blocks of audio data, one block typically representing 4, 8 or 16 milliseconds of data. The aggregation into such blocks of data provides a larger and more efficient unit for processing in the workstation. The size of block to use can be selected in accordance with conventional considerations (ie a trade-off between granularity and efficiency). Unlike the prior art however, a transmitted message will not contain a constant number of blocks. Typically each packet transmitted will include all the available audio data blocks, although in some cases it may be desirable, for example for reasons of queue integrity, to always maintain one block of data in the queue.

appropriate adapter card and software to allow communication over the network.

An embodiment of the invention will now be described by way of example with reference to the following drawings:

Figure 1 is a simplified schematic diagram of a computer system;

Figure 2 is a simplified diagram showing the major components of the adapter card of Figure 1;

Figures 3a and 3b schematically illustrate the queues of audio data at the transmitting and receiving stations respectively;

Figure 4 is a simplified flow chart illustrating the processing performed by the digital signal processor at the transmitting station;

Figure 5 is a simplified flow chart illustrating the processing performed by the system unit at the transmitting station; and

Figure 6 is a simplified flow chart illustrating the processing performed at the receiving station.

Figure 1 is a simplified schematic diagram of a computer system which may be used for audio transmission. The computer has a system unit 10 including microprocessor 22, semi-conductor memory (ROM/RAM) 24, and a bus over which data is transferred 26. Other conventional components of the computer (eg display, keyboard, mouse, etc) will also normally be present, but since they are not relevant to an understanding of the present invention they have been omitted from the drawings. The computer of Figure 1 may be any conventional workstation, such as an IBM PS/2 computer.

The computer of Figure 1 is equipped with two adapter cards. The first of these is a Token Ring adapter card 30. This card, together with accompanying software, allows messages to be transmitted onto and received from a Token Ring. The operation of the token ring card is well-known, and so again will not be described in detail. The second card is an audio card 28 which is connected to a microphone and a loudspeaker (not shown) for audio input and output respectiveley. The system of Figure 1 is typically used for two-way voice communications over a LAN, but may also be used in other multimedia applications, where one node in the network generates a sound signal (eg from an optical disk), which is transmitted over the network to be played out to a user at another node.

perfectly adequate.

The procedure at a terminal which generates audio data is as follows (see also the flow charts of Figures 4 and 5). The DSP aggregates 64 samples in the G711 format together into blocks of 64 bytes, corresponding to 8 ms of data (step 130). This represents the basic unit of processing and transmission over the network. The DSP then transfers these blocks into a circular buffer in main memory 24 (see Figure 3a) on the computer system (step 132) using direct memory access (DMA), in accordance with known techniques. Thus every 8 ms, a new block of data is written into memory. The DSP effectively maintains a record of the location of the last block added to memory, and increments this by 64 bytes for each new transfer. Finally, the DSP raises an interrupt (step 134) in a thread running on the main processor 22, again in accordance with known interrupt processing techniques, informing it that another block of data has been added to memory. The DSP cycles through the loop shown in Figure 4 every 8 ms.

In the current implementation, the interrupt in fact is concurrent with the DMA transfer, and actually signals the presence of the block added to memory in the previous 8 ms cycle. There is no direct check that the DMA for this previous cycle, which is an asynchronous process, has completed, since in normal circumstances the DMA takes far less than 8 ms to finish. Such a check could be added if desired, although due to the real-time nature of audio signals, it is difficult to see what corrective action could be taken even if a problem were detected.

Figure 3a represents the series of memory 24 locations into which blocks of audio data from the DSP are received (normally this will be a linear set of addresses, but is configured in software as a circular buffer using known programming techniques). A thread 150 executing on the main processor 22 receives the interrupt from the DSP (step 152), and uses this to update a pointer P1 to the location of the most recently added block of data (step 154). Thread 150 simply cycles through this process, keeping track of the contents in the circular buffer.

Another thread 160 concurrently running on the main processor 24 also executes in a continuous loop, requesting access (step 162) to the

facility, and play-out is to a user at another workstation, connected to the server via the network. Thus the sound can be transmitted from the server to the user's workstation using the approach described above.

Other modifications of the above system will also be readily apparent to the skilled person; for example, some form of data compression may be used to save bandwidth as the blocks are transmitted over the network, whether this be simple detection and excision of periods of silence, or the use of more sophisticated data compression techniques. Likewise, if the NETBIOS interface is not used (going either to a lower level interface, or perhaps a different type of network), there may be no need to specify the size of packet to be sent when network access is initially requested, but rather this can be determined at the time access to the network is actually obtained. In this case, a somewhat different processing strategy can be employed: each time access is granted, the difference between pointers P1 and P2 is calculated to determine the current number of audio blocks in the queue, and these are transmitted accordingly. P2 can then be updated to reflect the number of blocks sent without waiting for acknowledgement of their arrival.

9. The method of any preceding claim, further comprising the steps of: storing a queue of digital audio samples to be played out; receiving a data packet from the network containing a variable number of digital audio samples; and adding the received digital audio samples to the queue of digital audio samples to be played out.

10. A computer workstation comprising means for generating a sequence of digital audio samples for transmission over a computer network to a remote workstation; means for storing generated digital audio samples in a queue; means for obtaining access to the network; and means, responsive to access being obtained to the network, for transmitting a packet over the network containing essentially all the digital audio samples currently in the queue.

11. A computer workstation as claimed in claim 10, wherein the means for generating a sequence of digital audio samples comprises an audio adapter card.

12. A computer workstation as claimed in claim 11, wherein the audio adapter card accumulates the digital audio samples into audio data blocks, each containing a predetermined number of digital audio samples, and transfers one audio data block at a time into main memory of the workstation.

13. A computer workstation as claimed in claim 12, further comprising means for maintaining information indicating the number of audio data blocks in the queue, and means for updating said information each time a new audio data block is added to the queue or one or more audio data blocks are transmitted from the queue.

14. A computer workstation as claimed in claim 12 or 13, wherein the audio adapter card raises an interrupt in the computer workstation each time an audio data block is transferred into main memory.

15. A computer workstation as claimed in any of claims 10 to 14, wherein the means for transmitting comprises a local area network adapter card.